IN THE CLAIMS:

Please amend the claims as follows:

Claim 4 (Currently amended). Knowledge-driven architecture for control systems comprising: (i) semantic-enabled rules engine component or knowledgebase, further called knowledgebase, containing business domain ontology and business rules and application scenarios that reflect application requirements (ii) Application Scenario Player capable of transforming acts of scenarios and business rules into interactions with knowledgebase, presentation components, and the underlying application services (iii) Service Connector (iv) Presenter, (v) Service components with Controlled devices, drivers and mechanisms, where:

- the Application Scenario Player is connected via the Service Connector to the knowledgebase and application services and actively uses the knowledgebase in the process of application scenario interpretation;
- the Application Scenario Player interacts with the Service Connector that provides access to the knowledgebase and traditional services, as well as to the Presenter, which transforms resulting data into a proper presentation format;
- the Presenter receives data and presentation instructions from the Service Connector and interacts with multiple agents, providing one or more presentations options, including but not limited to video, audio, or electronic formats, wherein said presentation instructions may include definition-rules expressed by subject matter experts with business domain terms;

- wherein the Presenter can include but not limited to (a) Formatter that prepares data for audio or video interaction or for communication to other programs and passes data further to (b) Performer that uses formatted data for actual presentation to one or more agents via voice or screen or electronic formats for different types of agent devices;
- wherein the Service components receive controlling instructions from the Service Connector and transform them into specific signals and messages which Service components send to controlled devices, drivers and mechanisms including but not limited to navigation, 3-d movements, security and measurement systems;

Claim 5 (Previously presented). Knowledge-driven architecture of claim 4 further comprising: special input transformation components, including but not limited by speech, handwriting, and image recognition; wherein said input transformation components are connected and interact to the knowledgebase component filled with expected patterns and recognition scenarios and rules which provide transformation of multiple input types into traditional text and numeric variable values expected by event handling scenarios.

Claim 6 (Previously presented). Knowledge-driven architecture of claim 4 wherein the knowledge component further comprises a service adapter to the knowledgebase, providing a standard service interface required by the Service Connector, which interacts with the knowledgebase as with a set of services.

Claim 7 (Previously presented). Knowledge-driven architecture of claim 4 wherein the Service Connector component can include but is not limited to: (a) Object Retrieval that is able to find an existing service object or load the requested service class and instantiate the object at run-time (b) Object Registry that associates service objects with service and object names, stores service objects, and makes them reusable (c) Method Retrieval that retrieves the proper service method belonging to a selected service object based on the provided method arguments (d) Method Performer that performs the requested service operation on the selected service object.

Claim 8 (Previously presented). Knowledge-driven architecture of claim 4 further comprising the Optimizer component wherein the Optimizer takes a snapshot of existing rules and scenarios and translates them into a source code including but not limited by Java and C# languages, which can later be compiled into binary code to fix the current application rules into a regular application that lacks flexibility but provides better performance.

Claim 9 (Previously presented). Knowledge-driven architecture of claim 4 wherein the Scenario Player contains but is not limited to: (a) Input Type Checker that checks current input and, depending on its type, submits the input to one of several interpreters (b) One or more interpreters, including but not limited by the Scenario Act Interpreter, the Prompt Response Interpreter, the New Agent Request Interpreter, that, based on scenario and knowledgebase rules, translate

acts of scenarios or any other input into a direct action by one of the system components (c) Queue of Scenarios that stores the current scenario when it cannot be executed at the current time, but needs to be executed later (d) Success Analysis component that can store and retrieve a history of interpretation successes and failures, and in the case of interpretation failure invokes one of several learning scenarios that prompt an agent (a user or a program) to re-define the input or to provide more details for a better interpretation.

Claim 10 (Previously presented). Knowledge-driven architecture of claim 4 wherein service components are endowed with usage and value properties.

Claim 11 (Original). Knowledge-driven architecture of claim 9 wherein the Success Analysis component maintains and consistently refines a list of previously used services with their APIs, keywords, descriptions, and related scenarios in the knowledgebase, and re-evaluates the usage and value properties of the services.

Claim 12 (Original). Knowledge-driven architecture of claim 9 wherein the New Agent Request interpreter uses the list of previously used services with their APIs, keywords, descriptions, and related scenarios for automatic translation of user requests into service APIs and scenario acts.

Claim 13 (Original). Knowledge-driven architecture of claim 9 wherein the New Agent Request interpreter uses a list of previously used services with their APIs, keywords, descriptions, and related scenarios to offer selected parts of this information to the user for semi-automatic translation of new requests into service APIs and scenario acts.

Claim 14 (Previously presented). Knowledge-driven architecture of claim 4 further comprising the Communicator component wherein the Communicator provides collaborative access to knowledge and services existing on other distributed network systems built with this architecture.

Claim 15 (Original). Knowledge-driven architecture of claim 11 wherein the Success Analysis component propagates via the Communicator to distributed network systems information on a new service API or a new knowledge subject after the first success operation that included the service or the knowledge subject and then after each update provided locally by the Success Analysis component.

Claim 16 (Original). Knowledge-driven architecture of claim 15, wherein information on new elements is propagated after the first successful operation that included the new element, and thereafter after each local update of such information by the Success Analysis component.